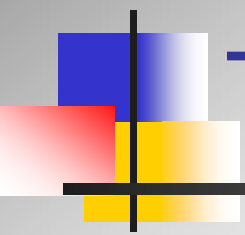
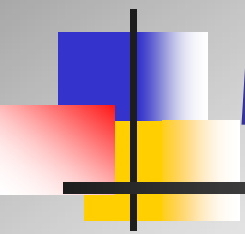


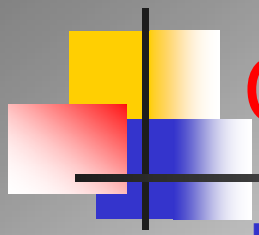
BG/L: Tuning for Many Nodes (Linpack)

John A. Gunnels
Mathematical Sciences Dept.
IBM T. J. Watson Research Center



BG/L: Tuning for Many Nodes (Linpack) *on a new machine that everyone wants for equally valid reasons*

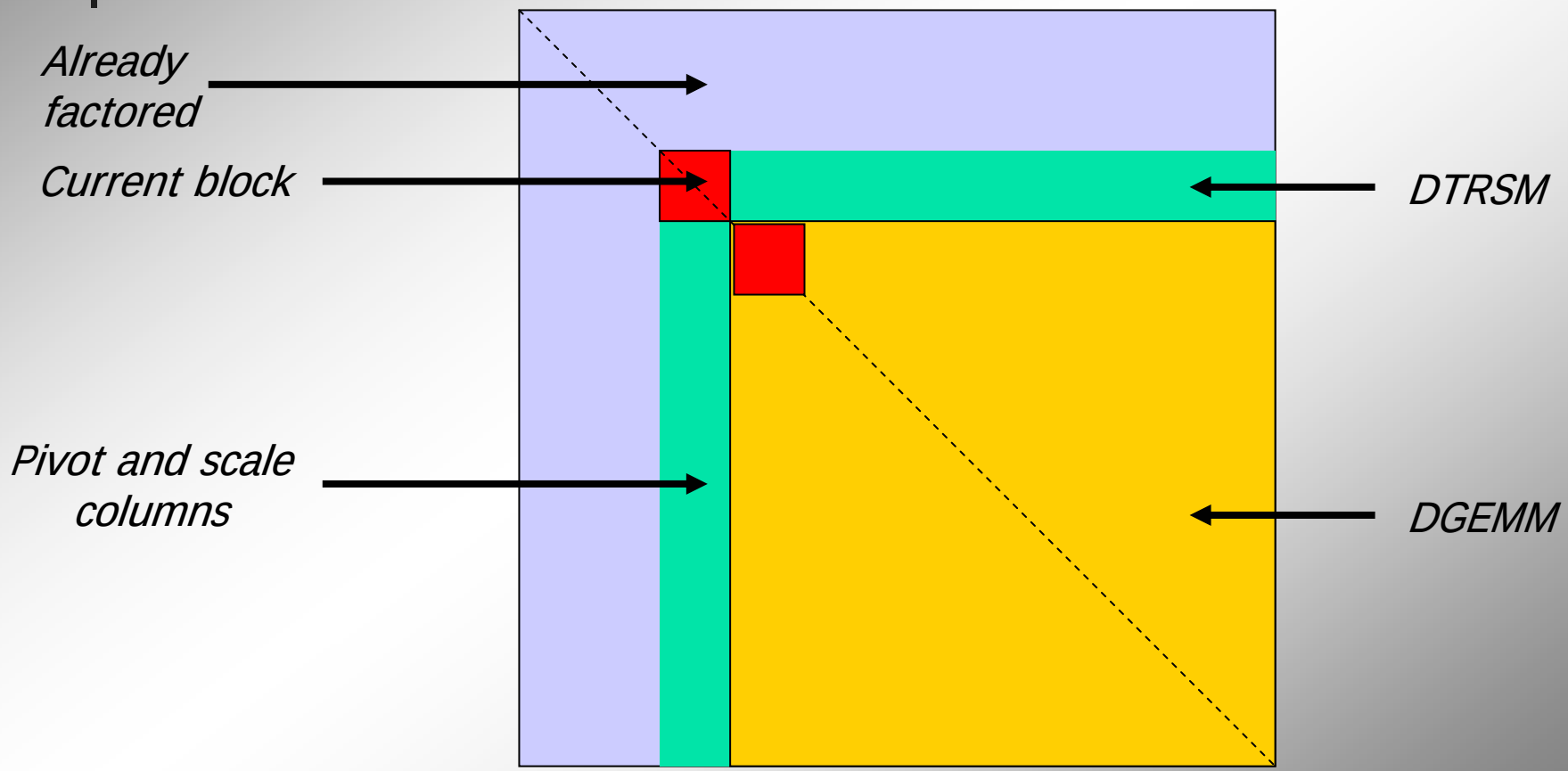
John A. Gunnels
Mathematical Sciences Department
IBM T. J. Watson Research Center



Overview

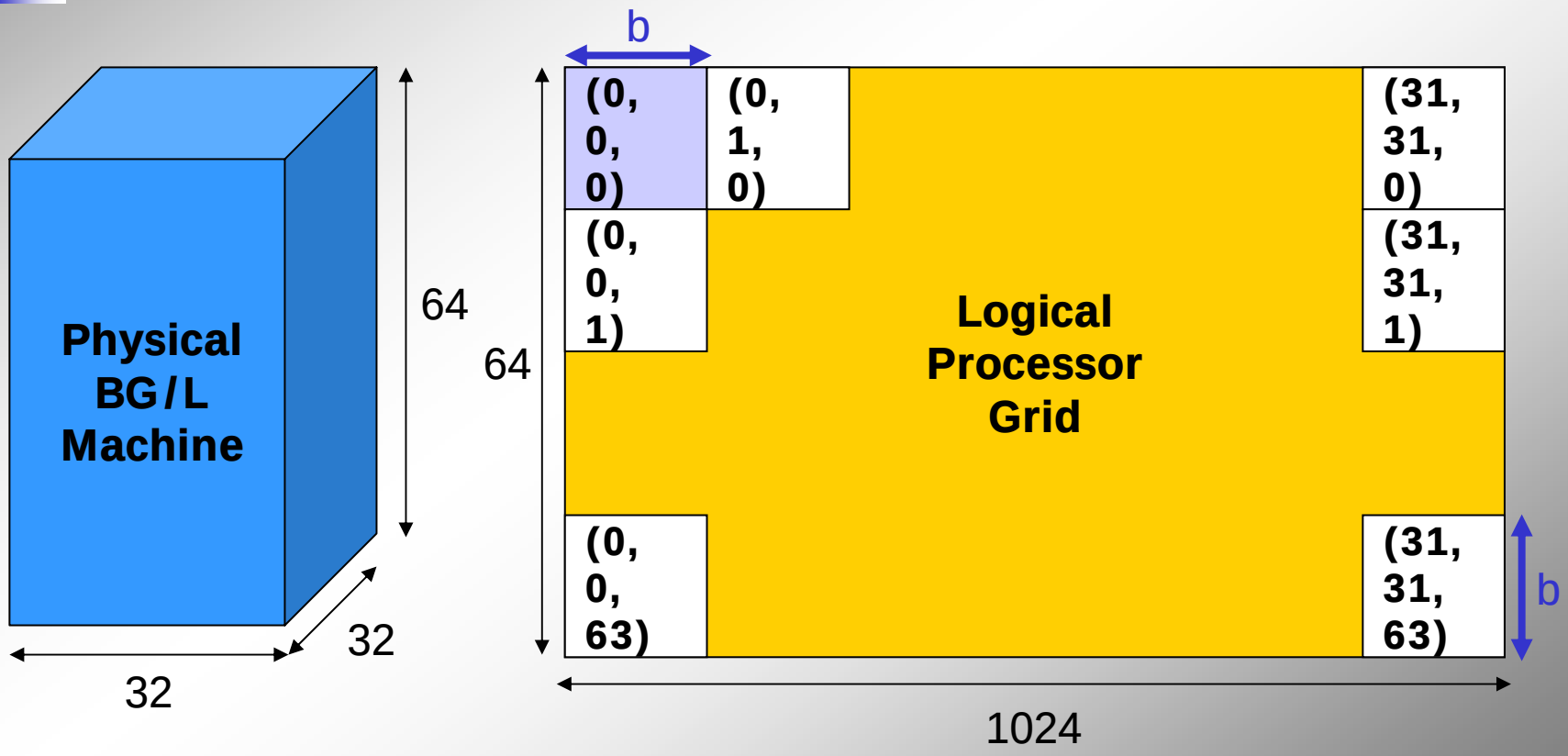
- Overcoming a computational bottleneck
- Problem Mapping
- Communication operations
  - Row broadcast
  - Column broadcast
  - Pivot identification
  - Pivot row exchange
- Linear algebra kernels (single node)
  - Matrix multiplication
  - Triangular solve, multiple RHS
  - Scaling, rank-1 updates (code fusion)

LU Factorization: Brief Review



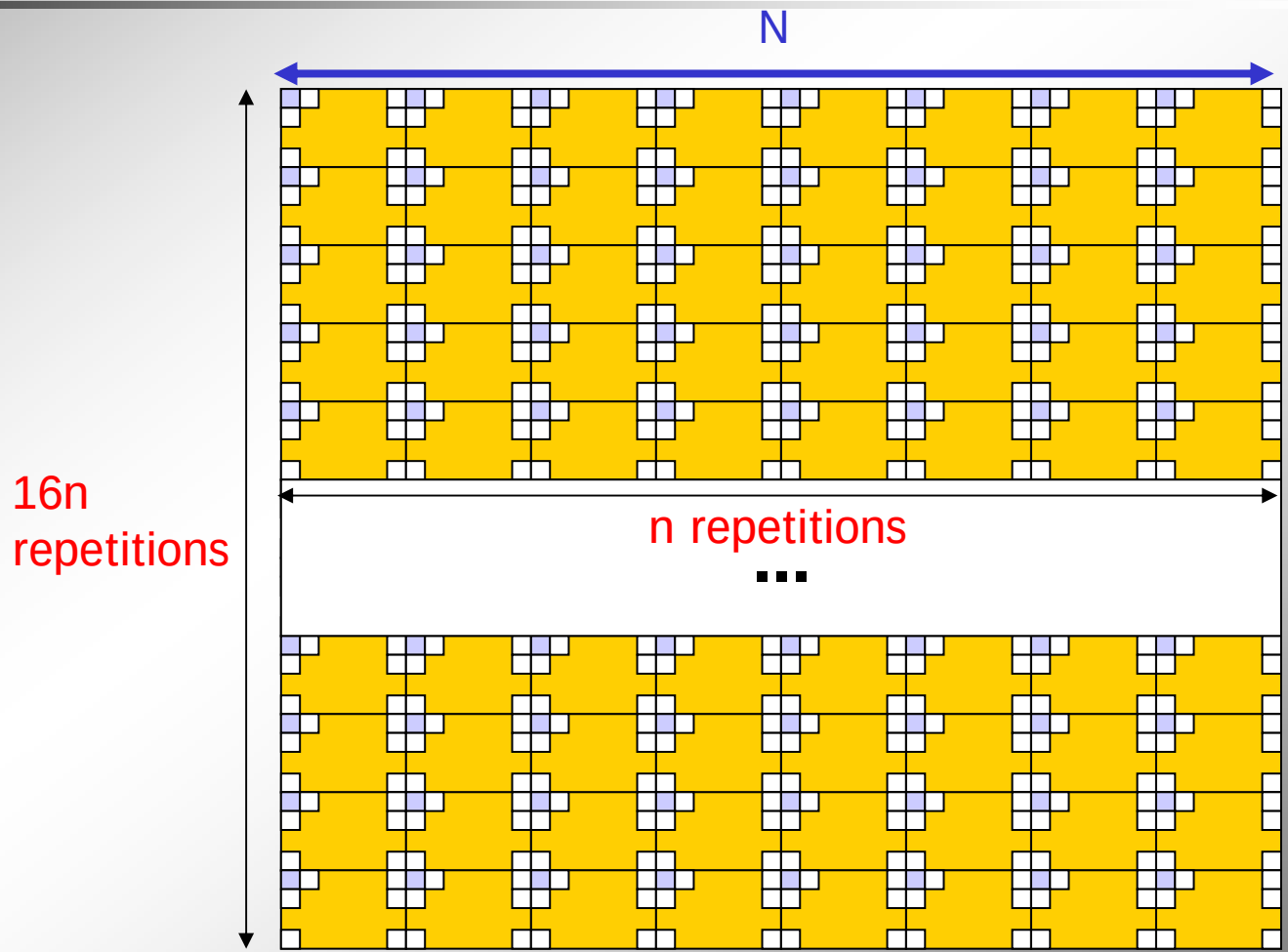
LINPACK

Physical to Logical Mapping

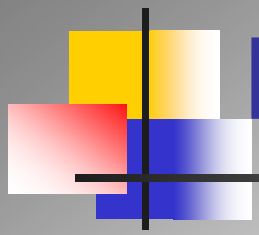


P(I,J,K) → **L(32I+J,K)**

LINPACK
Problem Mapping

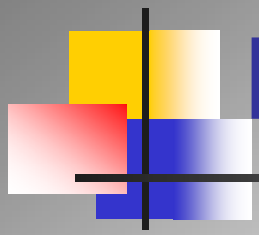


Panel Factorization: Option #1 Exploit Load Imbalance

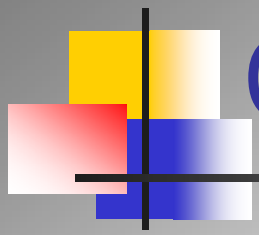


- Distributed over relatively few processors
  - Especially in the 64K node case
- May take as long as several DGEMM updates
  - Value may change as libraries change
- DGEMM load imbalance
  - Block size trades balance for speed
- Want to use collective communication primitives if possible
  - May require no “holes” in communication fabric (to achieve near-optimal performance)

Speed-up Option #2: Reduce Load Imbalance



- Change the data distribution
  - Decrease the critical path length
    - Speed up panel factorization
  - Take advantage of communication abilities of machine
- Complements Option #1
- Memory size (small favors #2; large #1)
  - Memory hierarchy (high latency: #1)
- The two options can be used in concert

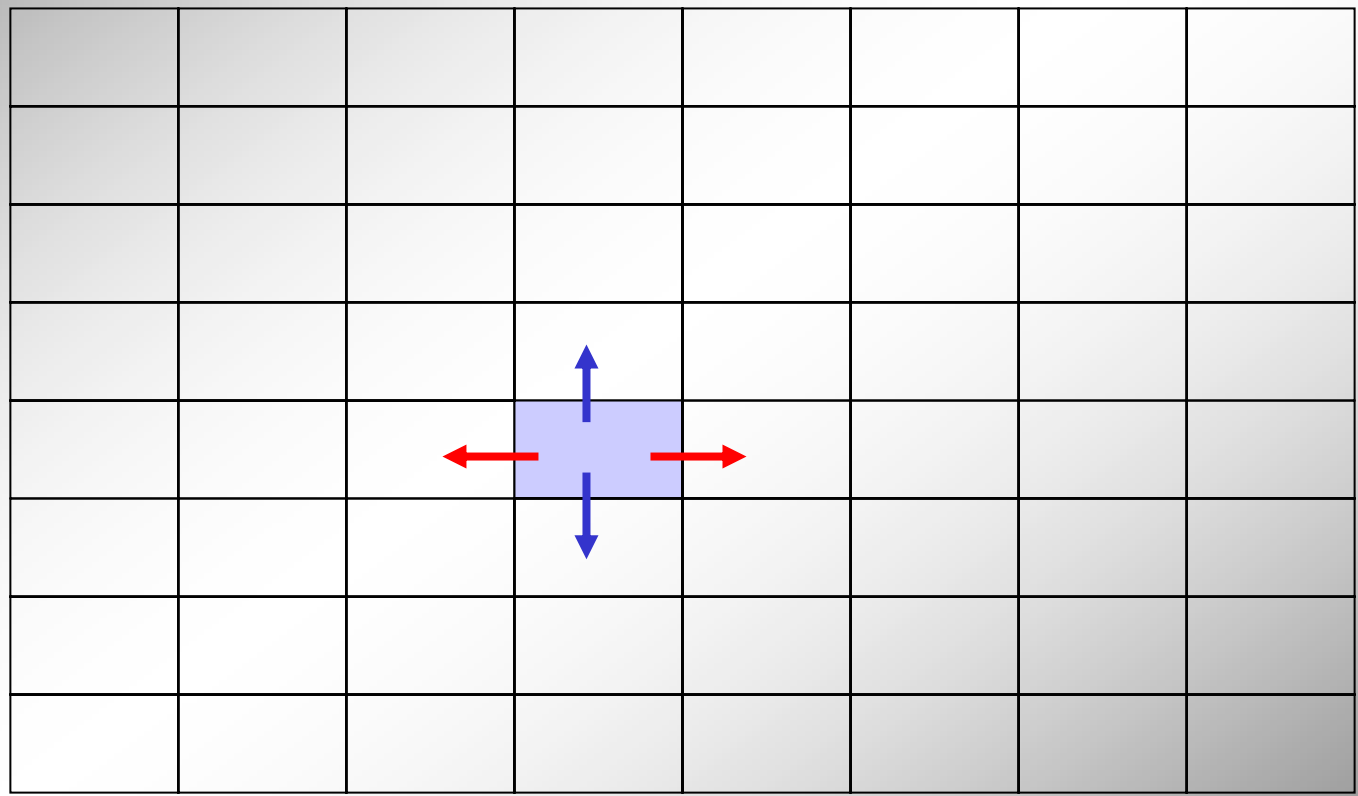


Communication Routines

- Broadcasts precede DGEMM update
- Routine needs to be architecturally aware
  - Multiple “pipes” connect processors
- Physical to logical mapping must be carefully managed
- Careful orchestration is required to take advantage of machines considerable abilities

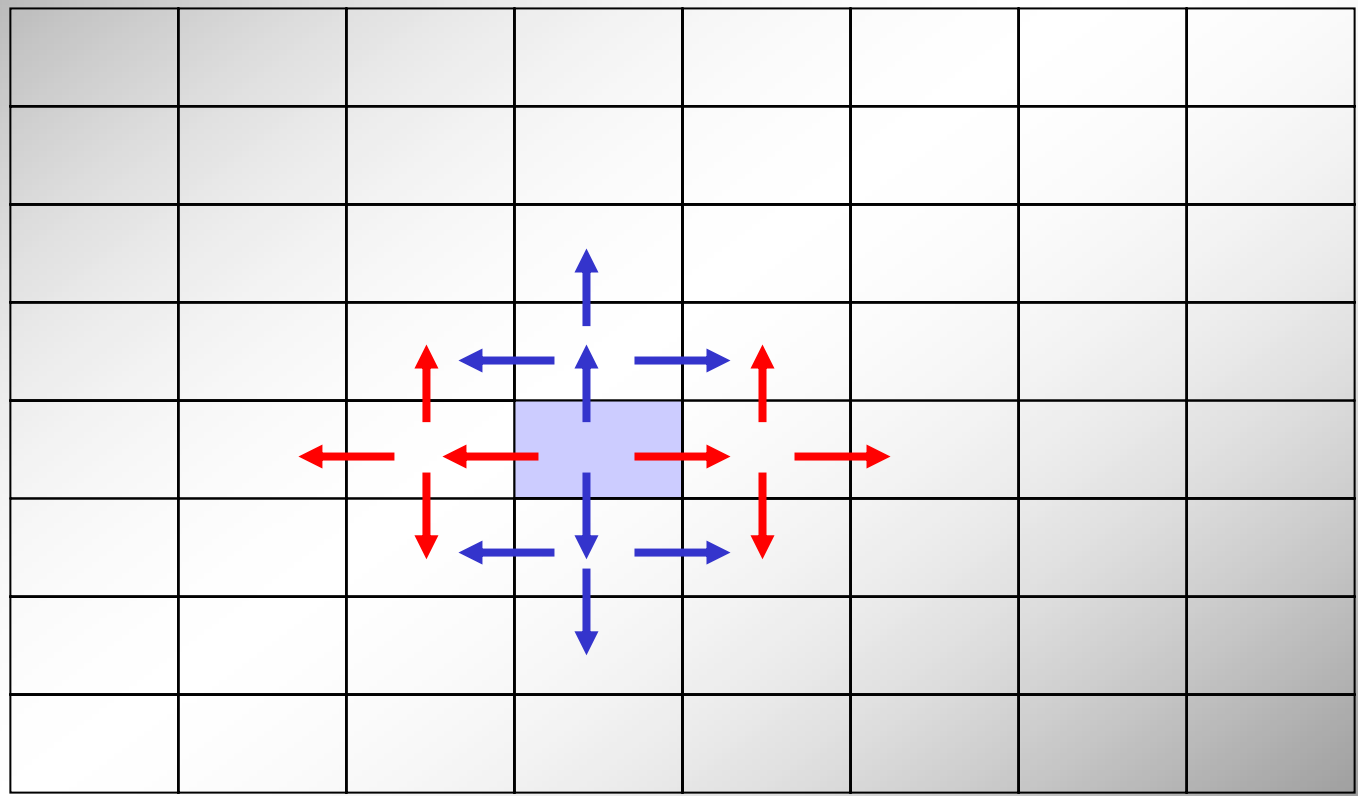
Row Broadcast

Mesh



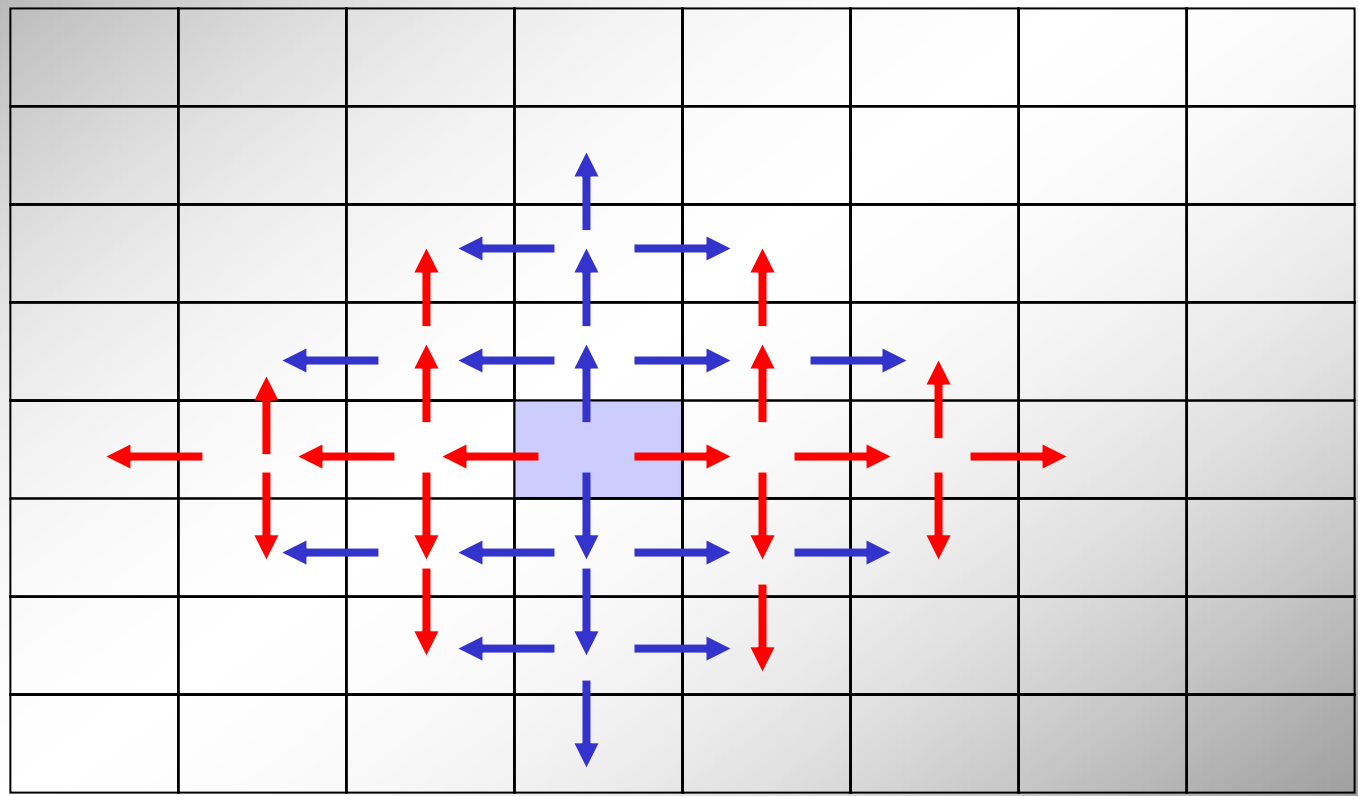
Row Broadcast

Mesh



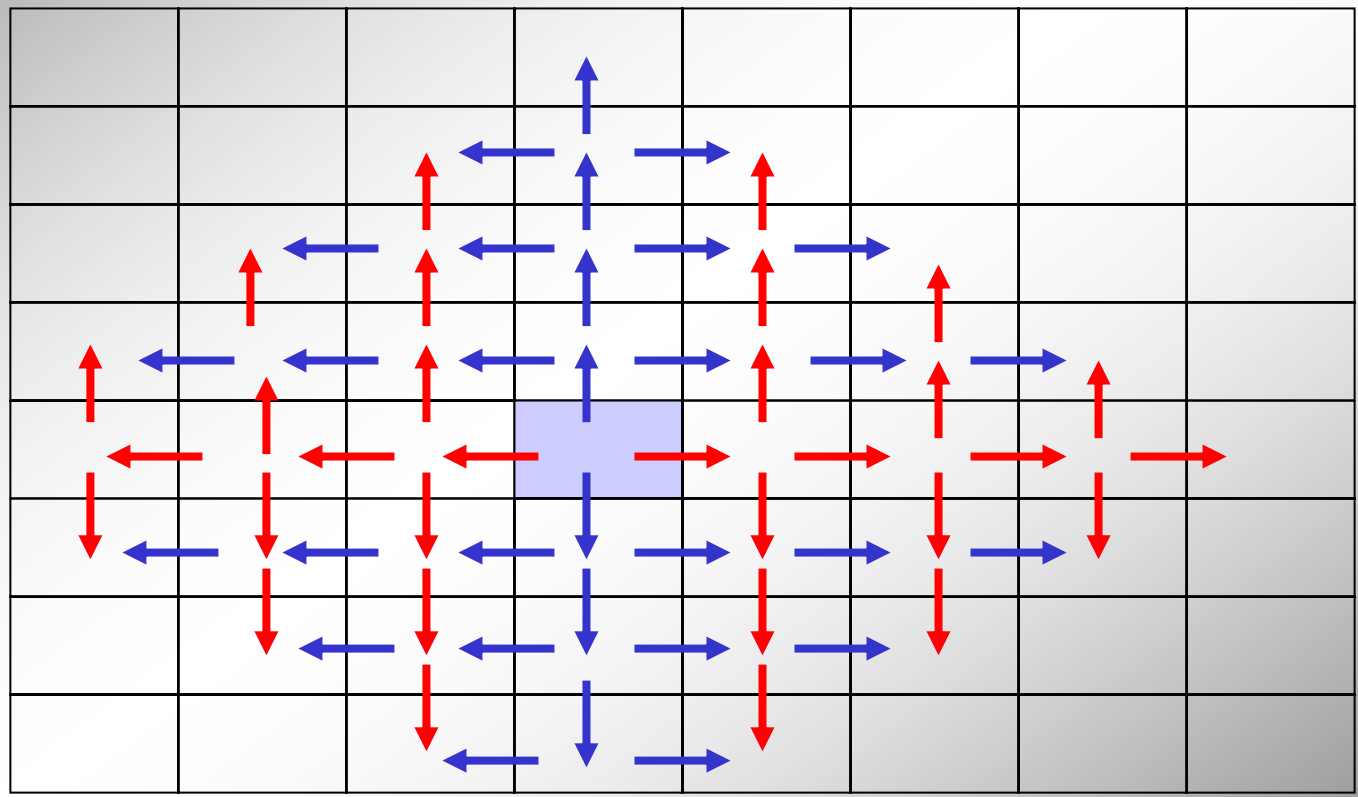
Row Broadcast

Mesh



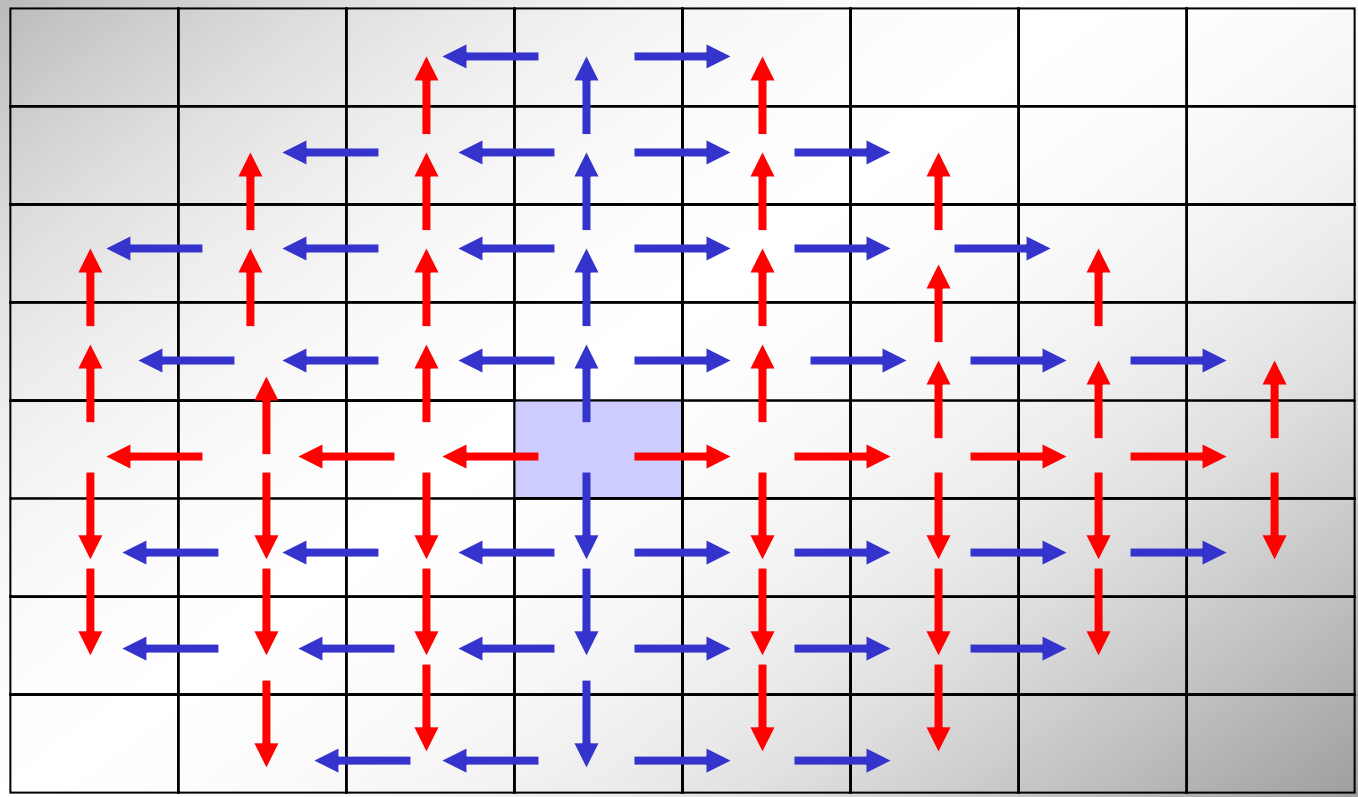
Row Broadcast

Mesh



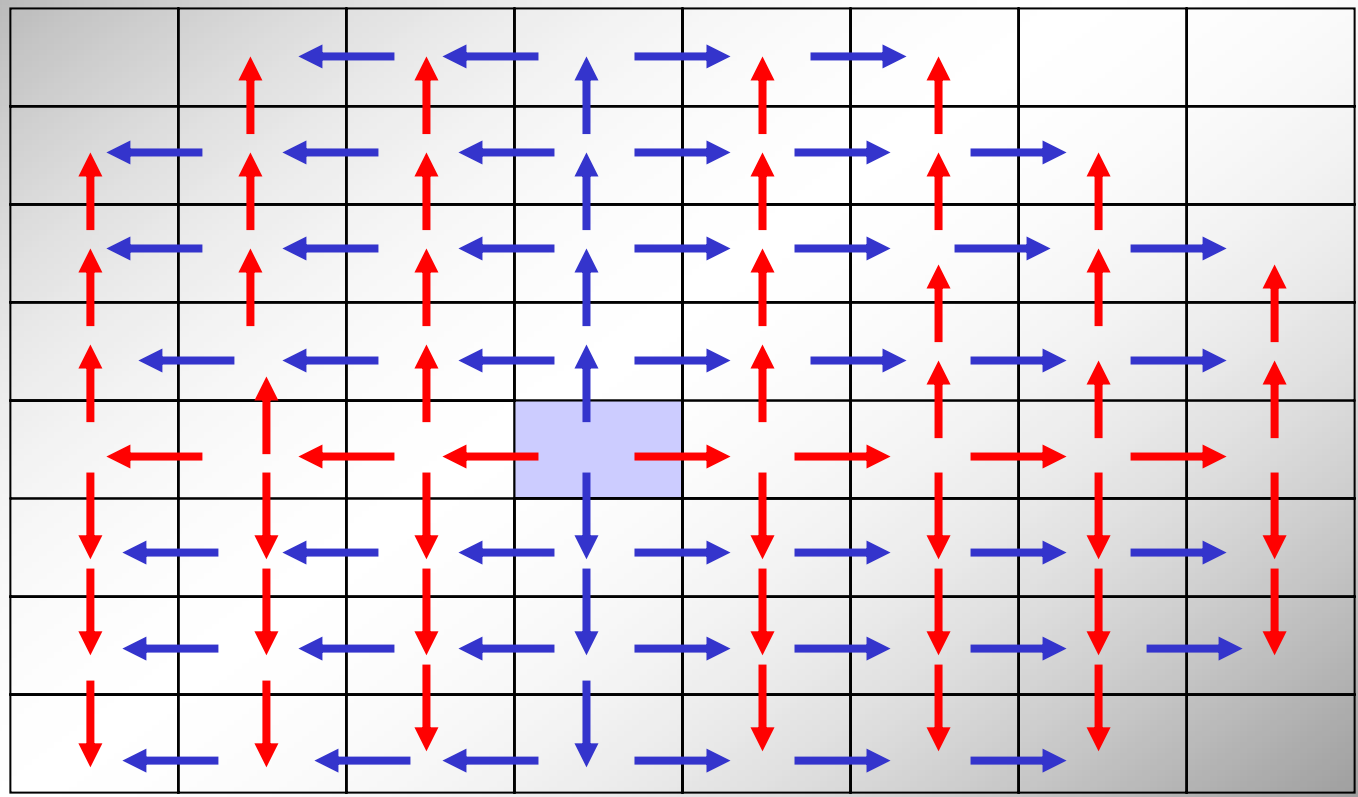
Row Broadcast

Mesh



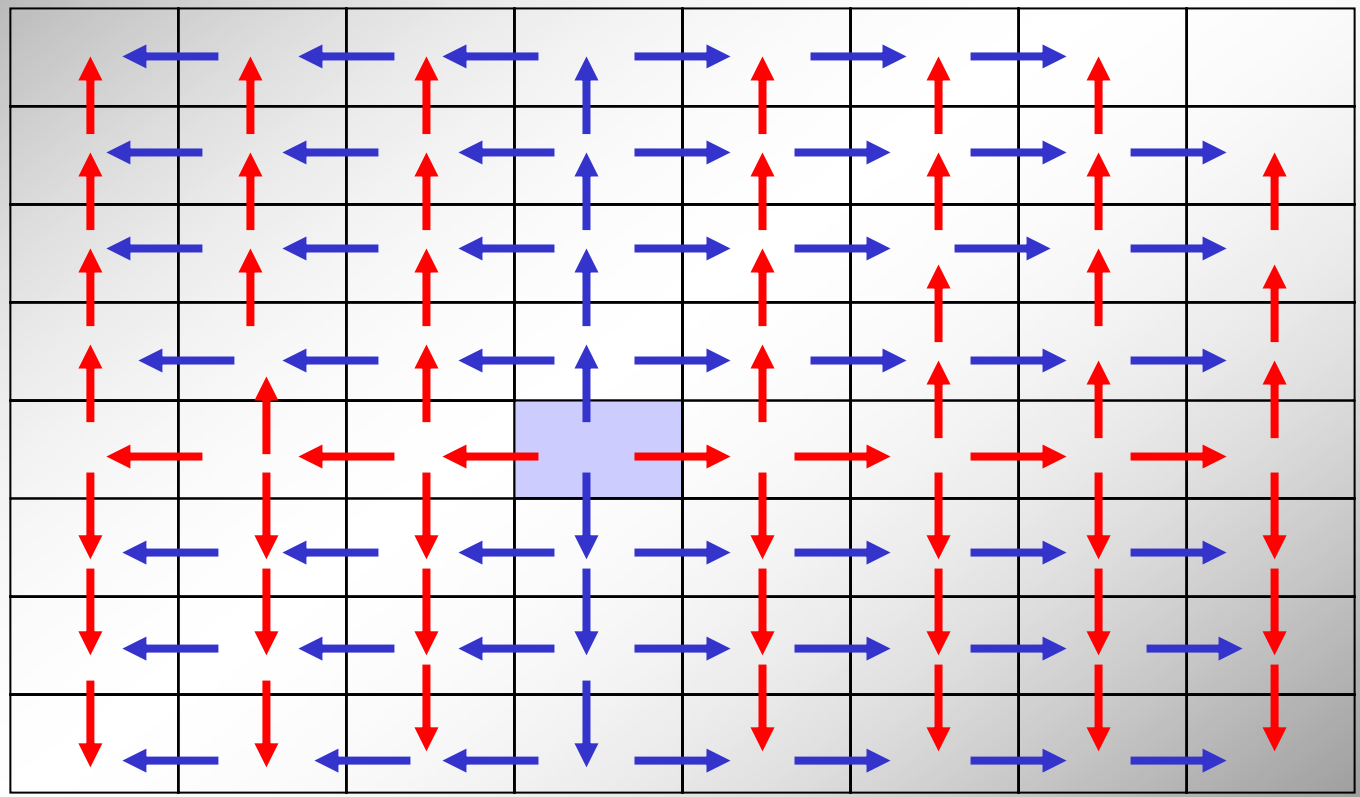
Row Broadcast

Mesh



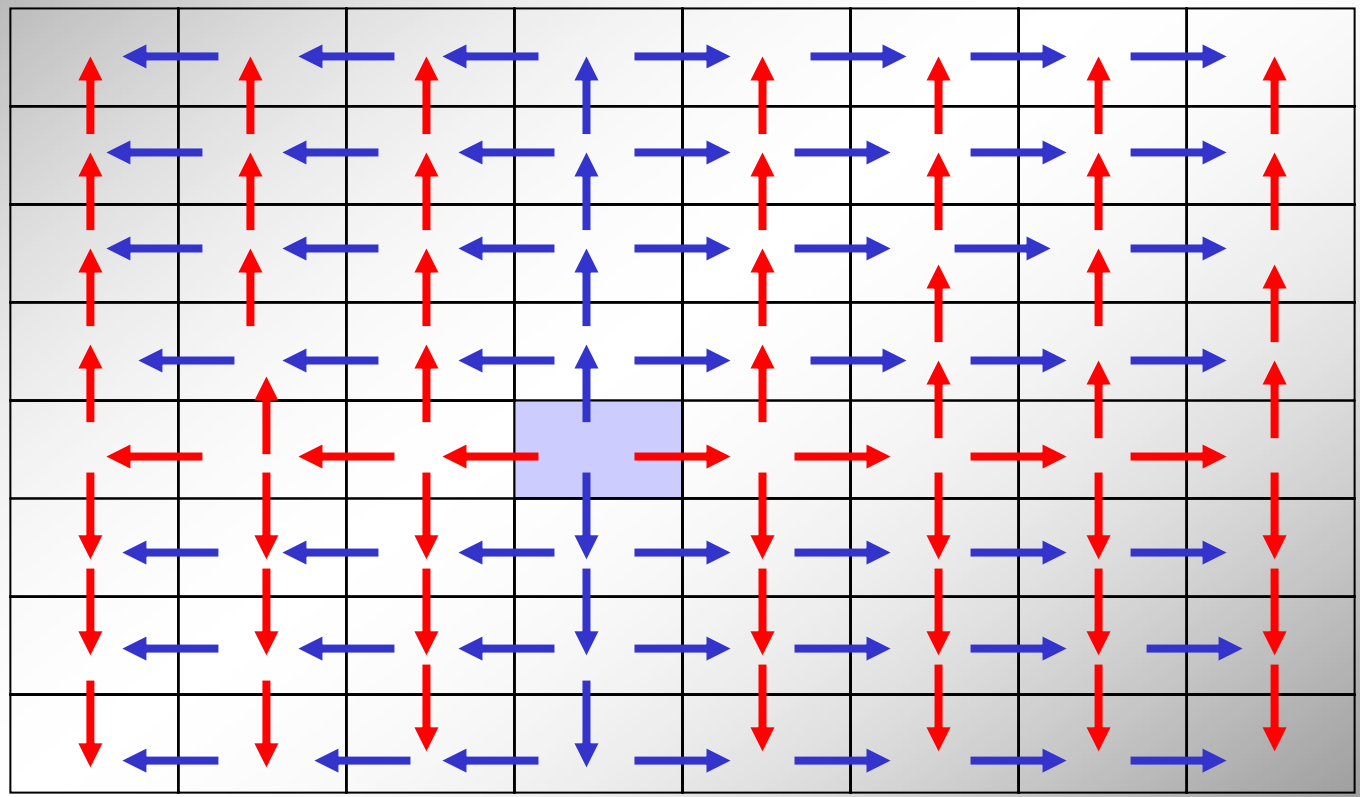
Row Broadcast

Mesh



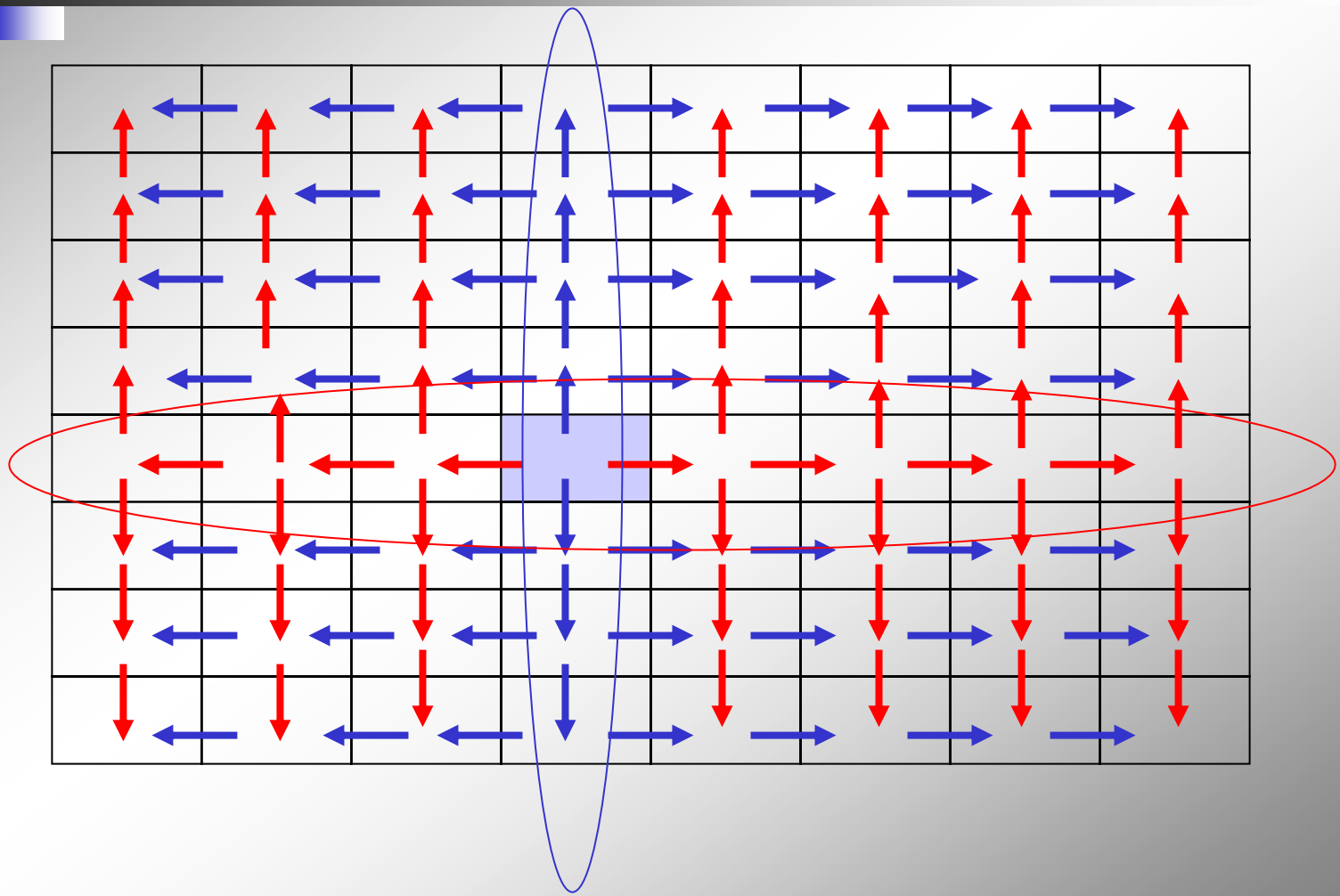
Row Broadcast

Mesh



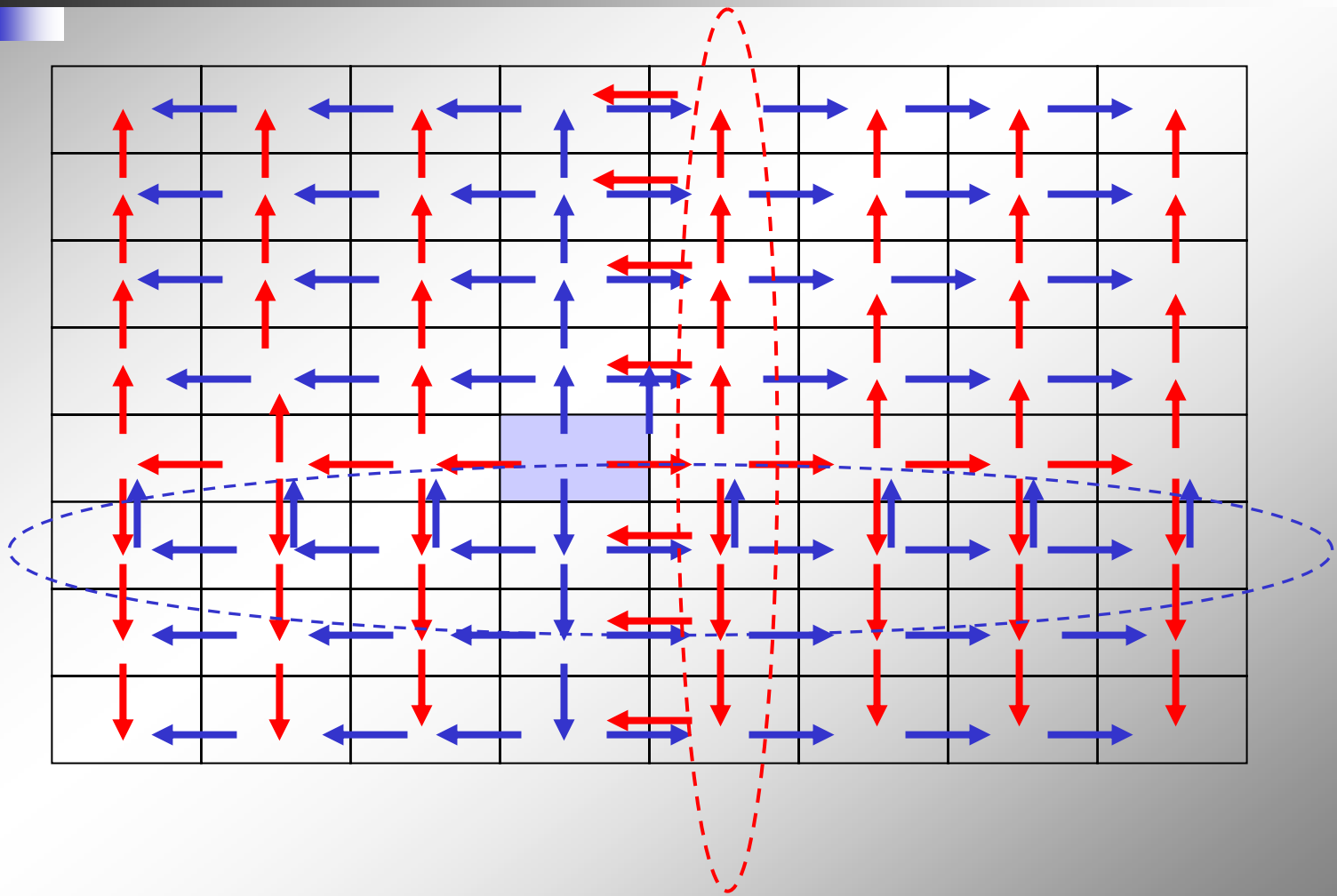
Row Broadcast

Mesh



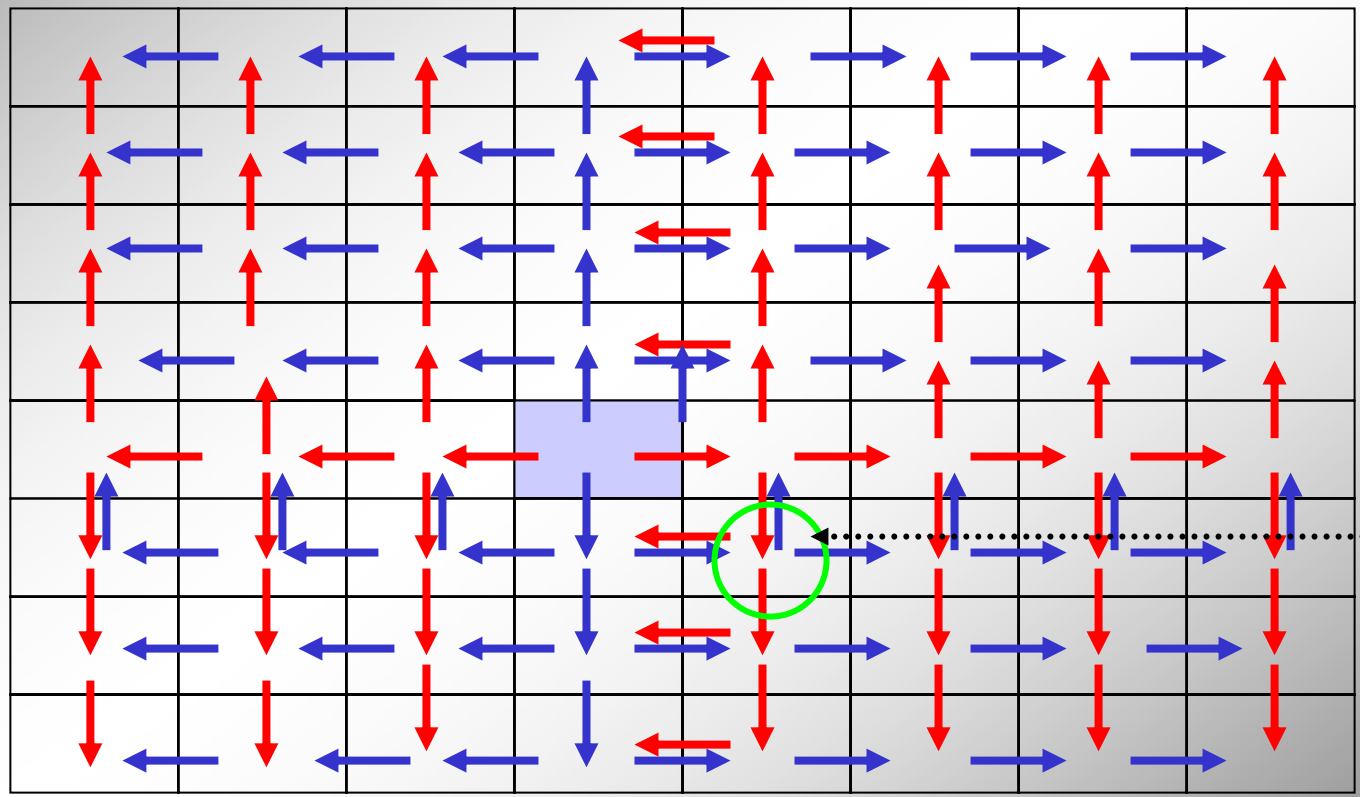
Row Broadcast

Mesh



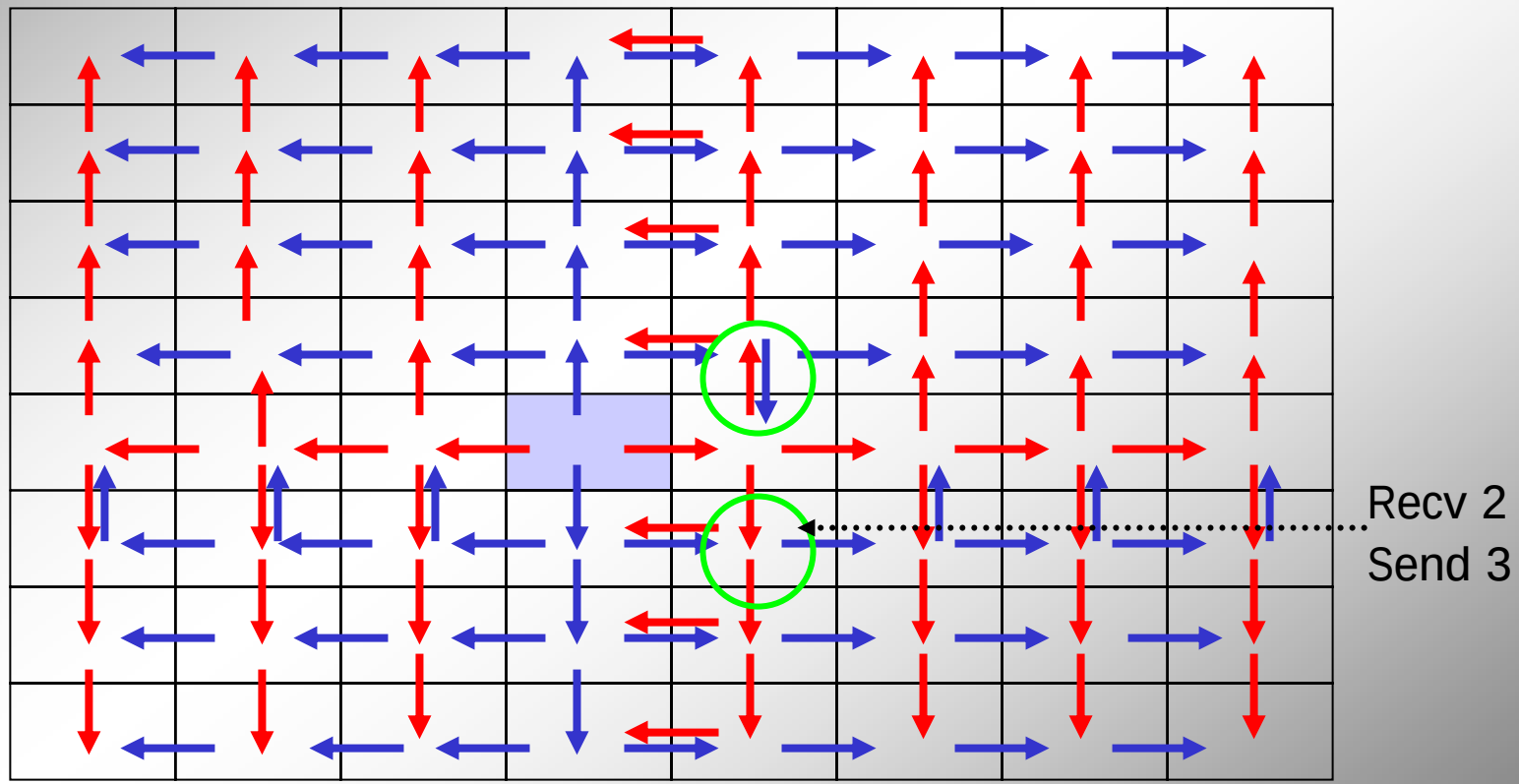
Row Broadcast

Mesh



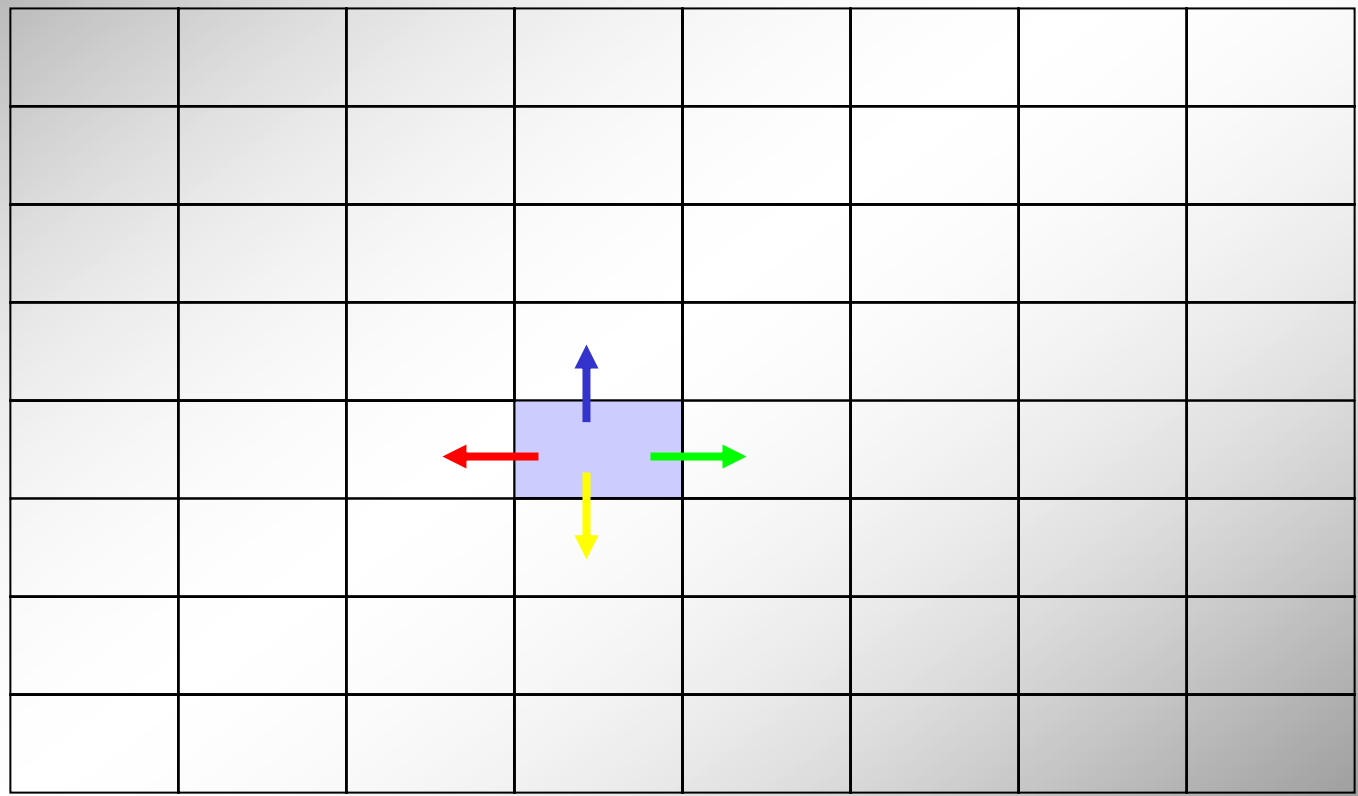
Row Broadcast

Mesh



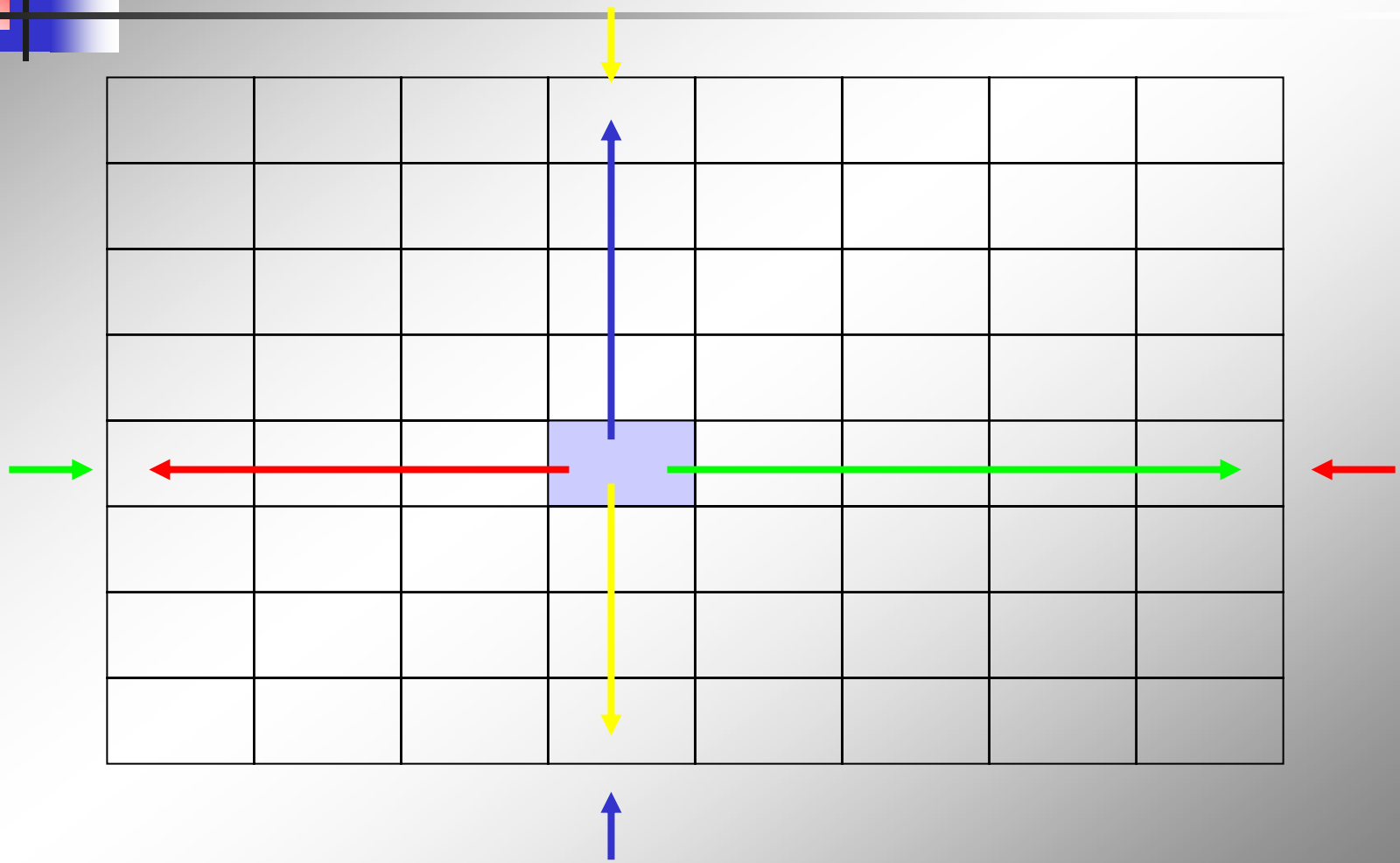
Row Broadcast

Torus



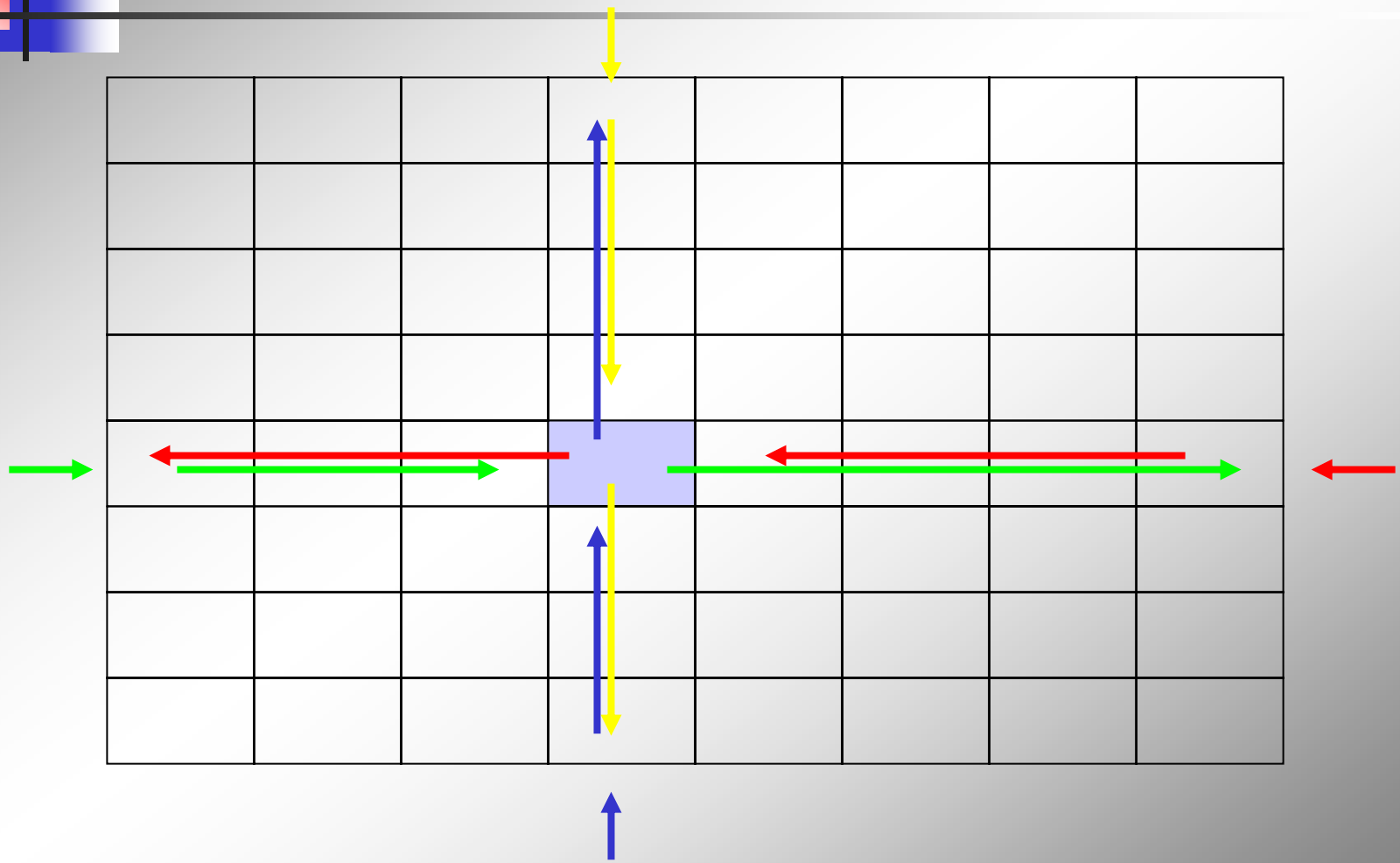
Row Broadcast

Torus



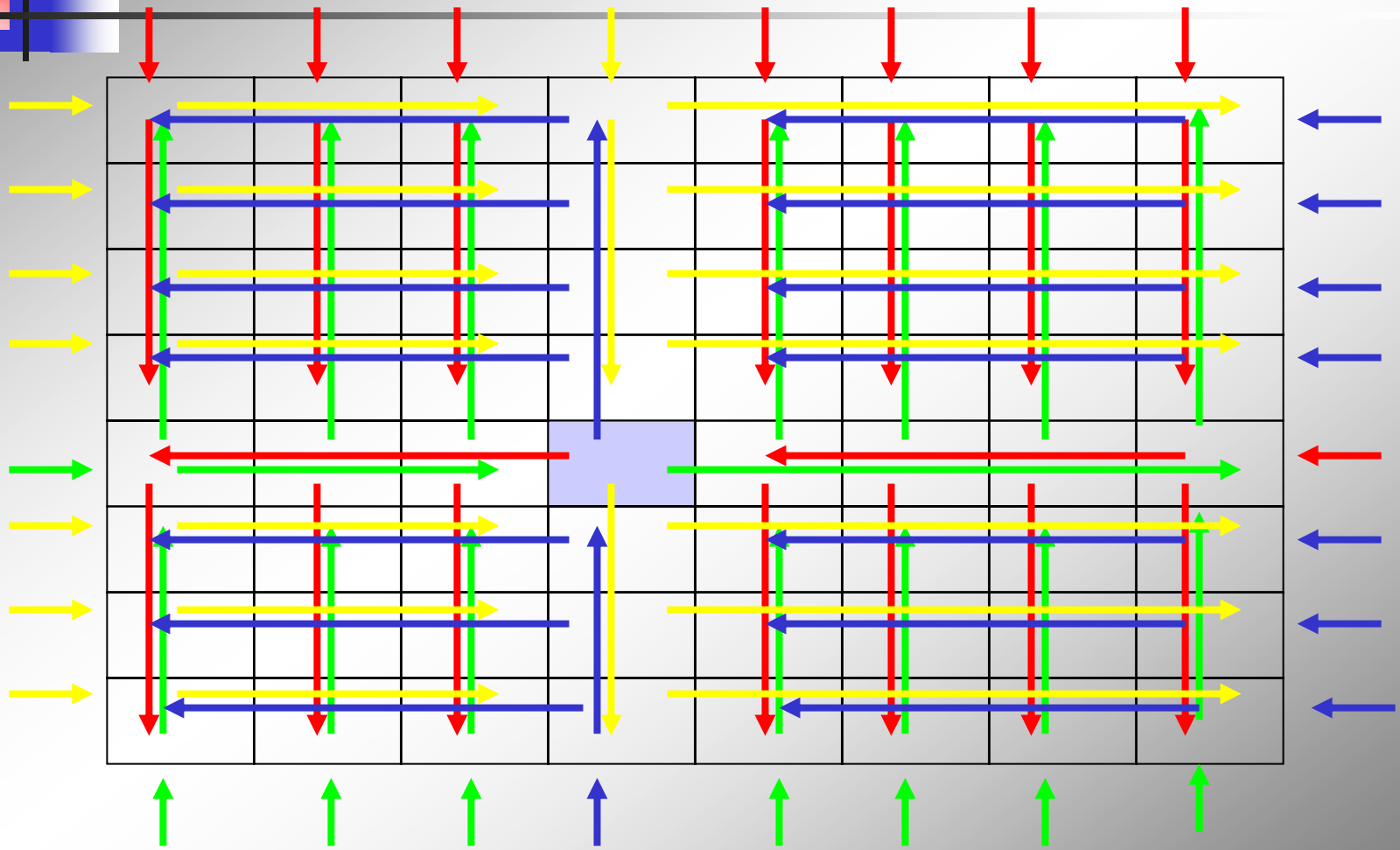
Row Broadcast

Torus



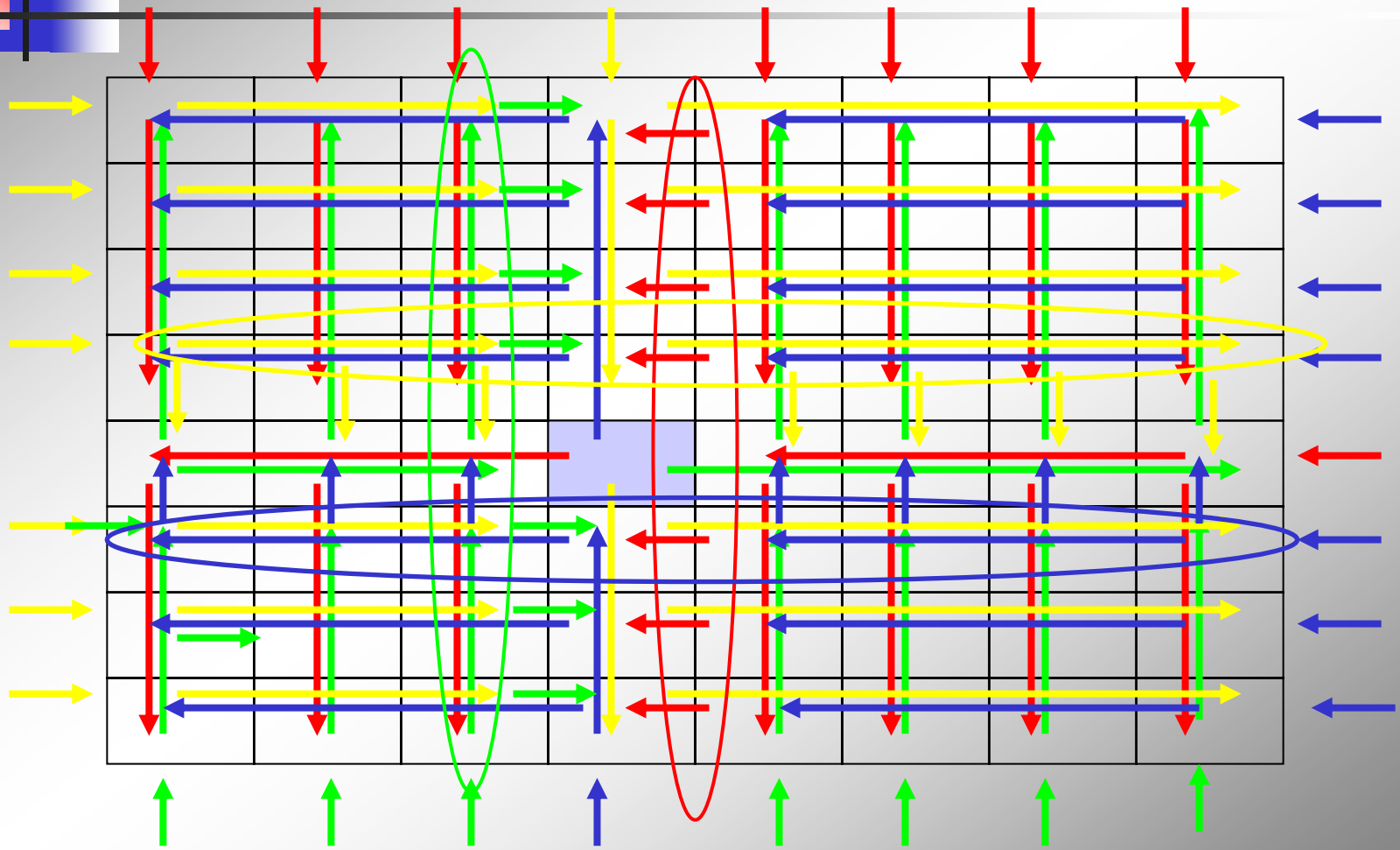
Row Broadcast

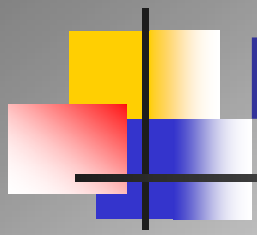
Torus



Row Broadcast

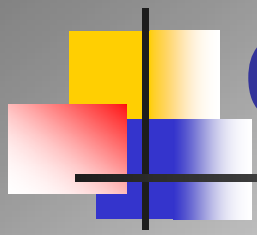
Torus





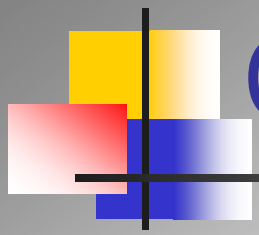
Broadcast

- Bandwidth/Latency
  - Bandwidth: 2 bytes/cycle per wire
  - Latency:
    - Sqrt(p), pipelined (large msg.)
    - Deposit bit: 3 hops
- Mesh
  - Recv 2/Send 3
- Torus
  - Recv 4/Send 4 (**no “hot spot”)**
  - Recv 2/Send 2 (red-blue only ... again, no bottleneck)
- Pipe
  - Recv/Send: 1/1 on mesh; 2/2 on torus



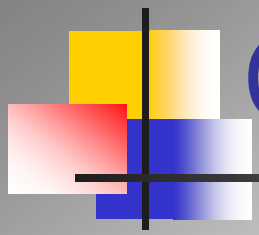
Conclusion

- Avoiding Bottlenecks
  - Overlapping differing computations
    - Beneficial on large or “memory walled” machines
  - Duplicating computations (local state)
    - Combine with moving from critical path
      - Or make “critical” less so
- Take advantage of hardware’s abilities
  - Algorithms & Architectures approach
    - “Mom & apple pie” (fundamental triangle; K-4)
  - Difference between good and optimal
    - Many characteristics are dynamic, but there is often a “safe” (fallback) method/approach/parameter



Conclusion

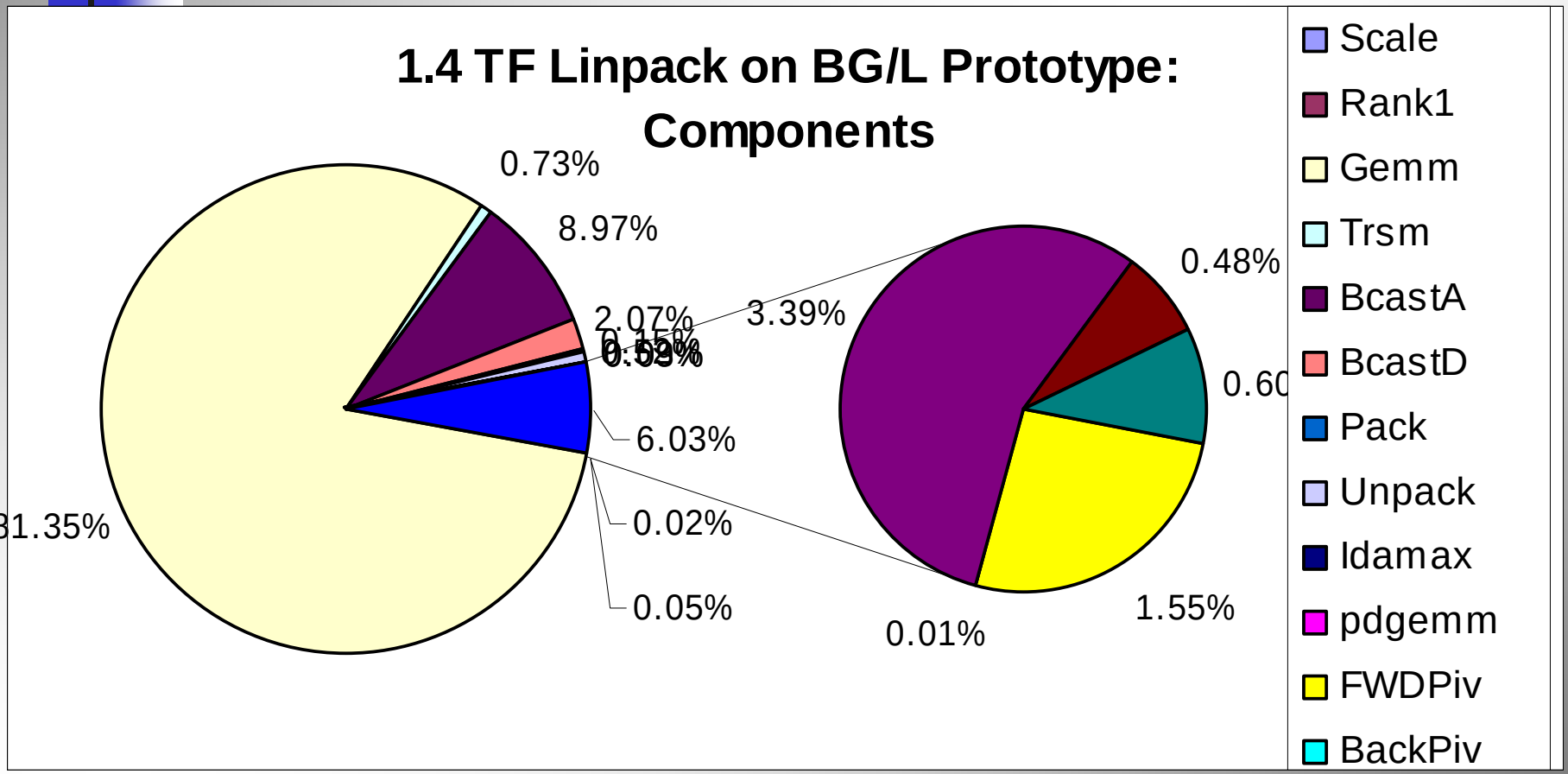
- Make use of models, extrapolated data
  - Use models to the extent that the architecture and algorithm are understood
  - Extrapolate from small processor sets
  - Vary as many (yes) parameters as possible at the same time
    - Consider how they interact and how they don't
    - Also remember that instruments affect timing
      - Often can compensate (incorrect answer results)
  - Utilize observed "eccentricities" with caution (MPI_Reduce)

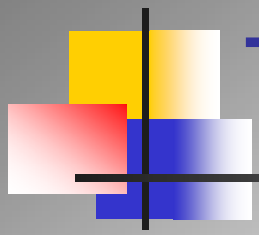


Conclusion

- Data Structures & Communications
  - Global: Altered distribution
    - Use as much hardware as possible
    - Have a path for software maturing process
  - Local: Recursive Data Formats (2nd talk)
    - Take advantage of local processor features
    - Large flops/memory ratio leads to an enter, re-map, execute, undo, exit pattern
  - Code fusion (2nd talk)

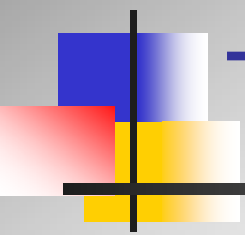
Conclusion II





Thanks to ...

- Gheorghe Almasi & Phil Heidelberger: MPI/Communications
- Vernon Austel: Data copy routines
- Gerry Kopcsay & Jose Moreira: System & machine configuration
- Derek Lieber & Martin Ohmacht: Refined memory settings
- Everyone else: System software & **Machine time!**



BG/L: Tuning for Many Nodes (Linpack)

John A. Gunnels
Mathematical Sciences Dept.
IBM T. J. Watson Research Center